

Serial No: 09/854,031

IN THE CLAIMS:

Please cancel claims 71-73, and amend claims 1, 7, 13, 15, 21, 27, 29, 32, 38, 44, 47, 50, 64-70, 74-80, 82, 89, 90 and 95 as follows:

1 Claim 1. (currently amended) A method comprising:
2 employing a computer for:
3 obtaining a collection of code;
4 providing a program graph representing said collection of code;
5 identifying any authorization resources of said collection of code;
6 locating any bounded path within said program graph; and
7 associating said any authorization resource with said any bounded path;
8 and
9 employing privileged code wherein a stop node represents a check
10 permission method and start nodes are any of root nodes in said program graph
11 or a node representing a privileged action method.

1 Claim 2. (original) A method as recited in claim 1, wherein said
2 collection of code includes codes obtained from a group of codes including
3 basic blocks, class methods, classes, collections of classes or any
4 combination of these.

1 Claim 3. (original) A method as recited in claim 1, wherein the step
2 of providing includes constructing said program graph through static analysis
3 techniques.

1 Claim 4. (original) A method as recited in claim 3, wherein the step
2 of constructing includes employing object code.

1 Claim 5. (original) A method as recited in claim 1, wherein the step
2 of identifying includes finding at least one authorization point in said
3 program graph.

Claim 6. (canceled)

1 Claim 7. (currently amended) A method as recited in claim 5, wherein
2 the step of finding the authorization point includes finding a
3 ~~AccessController.checkPermission~~ permission node in said program graph, and
4 finding a ~~java.security.Permission~~ permission object passed as an argument to
5 ~~the AccessController.checkPermission~~ a check permission method.

1 Claim 8. (original) A method as recited in claim 5, wherein said
2 authorization point is an instruction invocation.

Serial No: 09/854,031

1 Claim 9. (original) A method as recited in claim 8, wherein said
2 instruction invocation is used in a particular language for said collection
3 of code.

1 Claim 10. (original) A method as recited in claim 9, wherein said
2 particular language is C#.

1 Claim 11. (original) A method as recited in claim 1, wherein the step
2 of identifying includes employing data flow analysis.

1 Claim 12. (original) A method as recited in claim 11, wherein the step
2 of employing includes generating a data flow from said program graph.

1 Claim 13. (currently amended) A method as recited in claim 1, wherein
2 the step of identifying any bounded path includes locating a set of start
3 nodes in said program graph, and locating a the stop node in said program
4 graph; and said bounded path includes all nodes within the graph bound by
5 said start nodes and said stop node.

1 Claim 14. (original) A method as recited in claim 1, wherein the step
2 of associating includes associating and aggregating said any authorization
3 resource with said collection of code.

1 Claim 15. (currently amended) An apparatus comprising:
2 computing means having:
3 means for obtaining a collection of code;
4 means for providing a program graph representing said collection of
5 code;
6 means for identifying any authorization resources of said collection of
7 code;
8 means for locating any bounded path within said program graph; and
9 means for associating said any authorization resource with said any
10 bounded path; and
11 means for employing privileged code wherein a stop node represents a
12 check permission method and start nodes are any of root nodes in said program
13 graph or a node representing a privileged action method.

1 Claim 16. (original) An apparatus as recited in claim 15, wherein said
2 collection of code includes codes obtained from a group of codes including
3 basic blocks, class methods, classes, collections of classes or any
4 combination of these.

Serial No: 09/854,031

1 Claim 17. (original) An apparatus as recited in claim 15, wherein the
2 means for providing includes means for constructing said program graph
3 through static analysis techniques.

1 Claim 18. (original) An apparatus as recited in claim 17, wherein the
2 means for constructing includes employing object code.

1 Claim 19. (original) An apparatus as recited in claim 15, wherein the
2 means for identifying includes means for finding at least one authorization
3 point in said program graph.

Claim 20. (canceled)

1 Claim 21. (currently amended) An apparatus as recited in claim 19,
2 wherein the means for finding the authorization point includes finding a
3 ~~AccessController.checkPermission~~ permission node in said program graph, and
4 finding a ~~java.security.Permission~~ permission object passed as an argument to
5 ~~the AccessController.checkPermission~~ a check permission method.

1 Claim 22. (original) An apparatus as recited in claim 19, wherein said
2 authorization point is an instruction invocation.

1 Claim 23. (original) An apparatus as recited in claim 22, wherein said
2 instruction invocation is used in a particular language for said collection
3 of code.

1 Claim 24. (original) An apparatus as recited in claim 23, wherein said
2 particular language is C#.

1 Claim 25. (original) An apparatus as recited in claim 15, wherein the
2 means for identifying includes employing data flow analysis.

1 Claim 26. (original) An apparatus as recited in claim 25, wherein the
2 means for employing includes generating a data flow from said program graph.

1 Claim 27. (currently amended) An apparatus as recited in claim 15,
2 wherein the means for identifying any bounded path includes locating a set of
3 start nodes in said program graph, and locating ~~a~~ the stop node in said
4 program graph; and said bounded path includes all nodes within the graph
5 bound by said start nodes and said stop node.

1 Claim 28. (original) An apparatus as recited in claim 15, wherein the
2 means for associating includes associating and aggregating said any
3 authorization resource with said collection of code.

Serial No: 09/854,031

1 Claim 29. (currently amended) A method comprising:
 2 employing a computer including the steps of:
 3 obtaining a collection of code;
 4 providing a program graph representing said collection of code; and
 5 identifying a complete set of authorization resources of said
 6 collection of code; and
 7 employing privileged code wherein a stop node represents a check
 8 permission method and start nodes are any of root nodes in said program graph
 9 or a node representing a privileged action method.

1 Claim 30. (original) A method as recited in claim 29, if no resource
 2 is identified in said step of identifying, further comprising providing an
 3 indication that authorization testing is not necessary.

1 Claim 31. (original) A method as recited in claim 29, further
 2 comprising:
 3 identifying any bounded path within said program graph; and
 4 associating any authorization resource identified in the step of
 5 identifying with said any bounded path.

1 Claim 32. (currently amended) An apparatus comprising:
 2 a computer having access to a collection of code, the computer
 3 including:
 4 an authorization resource identifier to identify any authorization
 5 resources within the collection of code;
 6 a bounded path locator to locate any bounded path within a program
 7 graph of said collection of code; and
 8 an associator to associate said any authorization resource with said
 9 any bounded path; and
 10 wherein the bound path locator is configured to employ privileged code
 11 wherein a stop node represents a check permission method and start nodes are
 12 any of root nodes in said program graph or a node representing a privileged
 13 action method.

1 Claim 33. (original) An apparatus as recited in claim 32, wherein said
 2 collection of code includes codes obtained from a group of codes including
 3 basic blocks, class methods, classes, collections of classes or any
 4 combination of these.

1 Claim 34. (original) An apparatus as recited in claim 32, further
 2 comprising a program graph constructor to construct said program graph
 3 through static analysis techniques.

Serial No: 09/854,031

1 Claim 35. (original) An apparatus as recited in claim 32, wherein the
2 program graph constructor uses object code of said collection of code.

1 Claim 36. (original) An apparatus as recited in claim 32, wherein the
2 authorization resource identifier finds at least one authorization point in
3 said program graph.

Claim 37. (canceled)

1 Claim 38. (currently amended) An apparatus as recited in claim 36,
2 wherein the authorization point includes finding a
3 ~~AccessController.checkPermission~~ permission node in said program graph, and
4 finding a ~~java.security.Permission~~ permission object passed as an argument to
5 ~~the AccessController.checkPermission~~ a check permission method.

1 Claim 39. (original) An apparatus as recited in claim 36, wherein said
2 authorization point is an instruction invocation.

1 Claim 40. (original) An apparatus as recited in claim 39, wherein said
2 instruction invocation is used in a particular language for said collection
3 of code.

1 Claim 41. (original) An apparatus as recited in claim 40, wherein said
2 particular language is C#.

1 Claim 42. (original) An apparatus as recited in claim 32, wherein the
2 authorization resource identifier employs data flow analysis.

1 Claim 43. (original) An apparatus as recited in claim 42, wherein the
2 data flow analysis is generated from said program graph.

1 Claim 44. (currently amended) An apparatus comprising:
2 a computing module having access to a collection of code, the computer
3 module including:
4 an authorization resource identifier to completely identify any
5 authorization resources within a collection of code; and
6 a bounded path locator to locate any bounded path within a program
7 graph of said collection of code; and
8 wherein the bounded path locator is configured to employ privileged
9 code wherein a stop node represents a check permission method and start nodes
10 are any of root nodes in said program graph or a node representing a
11 privileged action method.

1 Claim 45. (original) An apparatus as recited in claim 44, wherein the
2 authorization resource identifier provides an indication that authorization

Serial No: 09/854,031

3 testing is not necessary if no resource is identified by said authorization
4 resource identifier.

1 Claim 46. (original) An apparatus as recited in claim 44, further
2 comprising an associator to associate said any authorization resource with
3 said any bounded path.

1 Claim 47. (currently amended) An apparatus comprising:
2 a computer including:
3 means for obtaining a collection of code;
4 means for providing a program graph representing said collection of
5 code; ~~and~~
6 means for identifying a complete set of authorization resources of said
7 collection of code; and
8 means for employing privileged code wherein a stop node represents a
9 check permission method and start nodes are any of root nodes in said program
10 graph or a node representing a privileged action method.

1 Claim 48. (original) An apparatus as recited in claim 47, further
2 comprising means for indicating if authorization testing is necessary.

1 Claim 49. (original) An apparatus as recited in claim 47, further
2 comprising:
3 means for identifying any bounded path within said program graph; and
4 means for associating any authorization resource identified in the step
5 of identifying with said any bounded path.

1 Claim 50. (currently amended) A method comprising:
2 employing a computer for:
3 constructing a program graph from a collection of code using static
4 analysis techniques;
5 performing a data flow analysis using static analysis techniques;
6 searching said program graph for any resource for executing said
7 collection of code;
8 identifying any bounded path within said program graph over which said
9 resource is utilized; ~~and~~
10 associating said resource with said collection of code;
11 wherein the step of associating includes mapping a subset of said
12 collection of code to said resource;
13 wherein the subset of said collection of code includes nodes in said
14 bounded path; and
15 employing privileged code wherein a stop node represents a check

Serial No: 09/854,031

16 permission method and start nodes are any of root nodes in said program graph
17 or a node representing a privileged action method.

1 Claim 51. (original) The method as recited in claim 50, wherein the
2 step of constructing includes employing source code of said collection of
3 code.

1 Claim 52. (original) A method as recited in claim 50, wherein the step
2 of constructing includes building an invocation graph of said collection of
3 code to form said program graph.

1 Claim 53. (original) A method as recited in claim 50, wherein the step
2 of constructing a program graph includes constructing a call graph of said
3 collection of code to form said program graph.

1 Claim 54. (original) A method as recited in claim 50, wherein said
2 step of constructing includes using context-sensitivity.

1 Claim 55. (original) A method as recited in claim 54, wherein said
2 step of using context sensitivity includes using type information for any
3 method receiver and/or any parameter.

1 Claim 56. (original) A method as recited in claim 55, wherein the step
2 of using type information includes using class and memory allocation site
3 information.

1 Claim 57. (original) A method as recited in claim 56, wherein the step
2 of using type information includes using per instance information.

1 Claim 58. (original) A method as recited in claim 57, wherein the step
2 of using instance information includes associating instance information with
3 a node or edge in said program graph.

1 Claim 59. (original) A method as recited in claim 50, wherein said
2 collection of code includes codes constructed from a group of codes including
3 basic blocks, class methods, classes, collections of classes or any
4 combination of these.

1 Claim 60. (previously presented) A method as recited in claim 50,
2 wherein the step of searching includes locating a node or edge in said
3 program graph that represents a location where said resource would be
4 utilized.

1 Claim 61. (previously presented) A method as recited in claim 60,
2 wherein said resource is a resource identifier.

Serial No: 09/854,031

1 Claim 62. (original) A method as recited in claim 60, wherein said
2 location is an authorization test.

1 Claim 63. (original) A method as recited in claim 50, wherein said
2 program graph represents an object oriented program.

1 Claim 64. (currently amended) A method as recited in claim 50, wherein
2 said program graph and said data flow analysis are constructed from ~~Java~~
3 object code.

1 Claim 65. (currently amended) A method as recited in claim 61, wherein
2 said resource identifier includes at least one ~~java.security.Permission~~
3 permission class.

1 Claim 66. (currently amended) A method as recited in claim 62, wherein
2 said authorization test is a call to any
3 ~~java.security.AccessController.checkPermission~~ check permission method.

1 Claim 67. (currently amended) A method as recited in claim 62, wherein
2 said location represents a call to any authorization testing method in any
3 instance of ~~java.lang.SecurityManager~~ a security manager class and/or one of
4 its subclasses.

1 Claim 68. (currently amended) A method as recited in claim 55, wherein
2 said node has a parameter which said type information is a
3 ~~java.security.Permission~~ permission class.

1 Claim 69. (currently amended) A method as recited in claim 68, wherein
2 the step of identifying includes locating the constructor for said
3 ~~java.security.Permission~~ permission class allocation site and using said data
4 flow analysis in identifying any value passed by any parameter to said
5 constructor, wherein the combination of the ~~Java java.security.Permission~~
6 permission class and a value for any parameter is the used ~~Permission~~
7 permission object.

1 Claim 70. (original) A method as recited in claim 50, wherein the step
2 of identifying any bounded path includes locating a set of start nodes in
3 said program graph, and locating ~~a~~ the stop node in said program graph; and
4 said bounded path includes all nodes within the graph bound by said start
5 nodes and said stop node.

1 Claim 71. (canceled)

1 Claim 72. (canceled)

Serial No: 09/854,031

1 Claim 73. (canceled)

1 Claim 74. (currently amended) A method as recited in claim ~~73~~ 50,
2 further comprising connecting a used ~~Permission~~ permission object with said
3 privileged code.

1 Claim 75. (currently amended) A method as recited in claim 74, further
2 comprising connecting a used ~~Permission~~ permission object with any node in
3 said program graph prior to ~~said~~
4 ~~java.security.AccessController.doPrivileged()~~ a privileged action node.

1 Claim 76. (currently amended) A method as recited in claim 74, wherein
2 said step of associating includes connecting a used ~~Permission~~ permission
3 object for each ~~java.security.AccessController.checkPermission()~~ check
4 permission method in said program graph.

1 Claim 77. (currently amended) A method as recited in claim 76, wherein
2 said step of associating includes connecting said used ~~Permission~~ permission
3 object from each node in said program graph to each method associated with
4 said program graph.

1 Claim 78. (currently amended) A method as recited in claim 77, wherein
2 said step of associating includes connecting said used ~~Permission~~ permission
3 object from each method in said program graph to each class in said program
4 graph.

1 Claim 79. (currently amended) A method as recited in claim 78, wherein
2 said step of associating includes connecting said used ~~Permission~~ permission
3 object from each class in said program graph to a collection of classes.

1 Claim 80. (currently amended) A method as recited in claim 79, wherein
2 said step of associating includes connecting said used ~~Permission~~ permission
3 object ~~to~~ includes associating associated from each class in said program
4 graph to a collection of classes.

1 Claim 81. (previously presented) A method as recited in claim 50,
2 further comprising employing said resource in executing said collection of
3 code.

1 Claim 82. (currently amended) A method comprising:
2 statically detecting resources for a collection of code written in a
3 computer programming language, by including the steps of:
4 calculating if any code of said collection of code is part of a program
5 graph;

Serial No: 09/854,031

6 identifying any resource for said collection of code;
7 determining any bounded path of nodes within said program graph which
8 constrain said resources of said collection of code; and
9 associating said any resource with said collection of code within said
10 bounded path of nodes in said program graph; and
11 employing privileged code wherein a stop node represents a check
12 permission method and start nodes are any of root nodes in said program graph
13 or a node representing a privileged action method.

1 Claim 83. (original) A method as recited in claim 82, wherein the step
2 of calculating includes using an invocation graph having a set of root nodes
3 for said program graph.

1 Claim 84. (original) A method as recited in claim 82, wherein the step
2 of calculating includes using a call graph having a set of root nodes for
3 said program graph.

1 Claim 85. (original) A method as recited in claim 83, further
2 comprising determining whether a basic block, method, class is represented as
3 a node in said invocation graph.

1 Claim 86. (original) A method as recited in claim 84, further
2 comprising determining whether a basic block, method, class is represented as
3 a node in said call graph.

1 Claim 87. (previously presented) A method as recited in claim 83,
2 further comprising identifying a node in said invocation graph that
3 identifies said resources, wherein said resources is one or more resources.

1 Claim 88. (previously presented) A method as recited in claim 84,
2 further comprising a step identifying a node in said invocation graph that
3 identifies said resources, wherein said resources is one or more resources.

1 Claim 89. (currently amended) A method as recited in claim 83, wherein
2 nodes in said bounded path of nodes are all nodes in the graph that are
3 bounded between start nodes and a the stop node.

1 Claim 90. (currently amended) A method as recited in claim 84, wherein
2 nodes in said bounded path of nodes are all nodes in the graph that are
3 bounded between a set of start nodes and a the stop node.

1 Claim 91. (original) A method as recited in claim 90, wherein all root
2 nodes in said invocation graph are members of the start nodes.

Serial No: 09/854,031

1 Claim 92. (original) An article of manufacture comprising a computer
2 usable medium having computer readable program code means embodied therein
3 for causing association, the computer readable program code means in said
4 article of manufacture comprising computer readable program code means for
5 causing a computer to effect the steps of claim 1.

1 Claim 93. (original) An article of manufacture comprising a computer
2 usable medium having computer readable program code means embodied therein
3 for causing identification, the computer readable program code means in said
4 article of manufacture comprising computer readable program code means for
5 causing a computer to effect the steps of claim 29.

1 Claim 94. (original) An article of manufacture comprising a computer
2 usable medium having computer readable program code means embodied therein
3 for causing association, the computer readable program code means in said
4 article of manufacture comprising computer readable program code means for
5 causing a computer to effect the steps of claim 50.

1 Claim 95. (currently amended) An article of manufacture comprising a
2 computer usable medium having computer readable program code means embodied
3 therein for causing statistical detection, the computer readable program code
4 means in said article of manufacture comprising computer readable program
5 code means for causing a computer to effect the steps of claim ~~80~~ 83.

1 Claim 96. (original) A computer program product comprising a computer
2 usable medium having computer readable program code means embodied therein
3 for causing association, the computer readable program code means in said
4 computer program product comprising computer readable program code means for
5 causing a computer to effect the functions of the elements in claim 15.

1 Claim 97. (original) A computer program product comprising a computer
2 usable medium having computer readable program code means embodied therein
3 for causing association, the computer readable program code means in said
4 computer program product comprising computer readable program code means for
5 causing a computer to effect the functions of the elements in claim 32.

1 Claim 98. (original) A computer program product comprising a computer
2 usable medium having computer readable program code means embodied therein
3 for causing identification, the computer readable program code means in said
4 computer program product comprising computer readable program code means for
5 causing a computer to effect the functions of the elements in claim 44.

Serial No: 09/854,031

1 Claim 99. (original) A computer program product comprising a computer
2 usable medium having computer readable program code means embodied therein
3 for causing identification, the computer readable program code means in said
4 computer program product comprising computer readable program code means for
5 causing a computer to effect the functions of the elements in claim 47.

1 Claim 100. (original) A program storage device readable by machine,
2 tangibly embodying a program of instructions executable by the machine to
3 perform method steps for identification, said method steps comprising the
4 steps of claim 1.